

AMENDMENTS TO THE CLAIMS

Please amend the claims as follows.

1. (Currently Amended) A method for storing [[a]] data blocks, comprising:
storing [[the]] a first data block and a second data block in a storage pool;
obtaining a first data block location and a second data block location;
calculating a first data block checksum for the first data block;
calculating a second data block checksum for the second data block; and
storing a first indirect block referencing the first data block and the second data block in
the storage pool, wherein the first indirect block comprises a first block pointer
comprising the first data block location and the first data block checksum and a
second block pointer comprising the second data block location and the second
data block checksum.
2. (Currently Amended) The method of claim 1, further comprising:
calculating a first indirect block checksum for the first indirect block;
obtaining a first indirect block location; and
storing a second indirect block in the storage pool, wherein the second indirect block
comprises the first indirect block location and the first indirect block checksum.
3. (Original) The method of claim 1, further comprising:
assembling the first indirect block.
4. (Currently Amended) The method of claim 3, wherein assembling the first indirect block
comprises:
storing the first data block checksum in a checksum field in [[a]] the first block pointer in
the first indirect block, and
storing the first data block location in the first block pointer, wherein storing the data
block location comprises storing a metaslab ID and offset.
5. (Currently Amended) The method of claim 4, further comprising:
storing a birth value in a birth field in the first block pointer.

6. (Original) The method of claim 3, wherein the first indirect block is assembled using a data management unit.
7. (Original) The method of claim 1, wherein the storage pool comprises at least one storage device.
8. (Original) The method of claim 1, wherein the storage pool is divided into a plurality of metaslabs.
9. (Original) The method of claim 8, wherein each of the plurality of metaslabs is associated with a metaslab ID.
10. (Currently Amended) The method of claim 9, wherein the first data block location comprises the metaslab ID and an offset.
11. (Currently Amended) The method of claim 1, wherein storing the first data block and the second data block comprises using a storage pool allocator.
12. (Original) A method for storing a first data block and a second data block, comprising:
 - storing the first data block and the second data block in a storage pool;
 - obtaining a first data block location and a second data block location;
 - calculating a first data block checksum for the first data block;
 - calculating a second data block checksum for the second data block; and
 - storing an array of block pointers in an indirect block, wherein the array block of pointers comprises,
 - a first block pointer comprising the first data block location and the first data block checksum, and
 - a second block pointer comprising the second data block location and the second data block checksum.
13. (Original) The method of claim 12, wherein the indirect block is assembled using a data management unit.
14. (Original) The method of claim 12, wherein the indirect block is stored using a storage pool allocator.

15. (Currently Amended) A method for retrieving data in a data block, comprising:
- obtaining an indirect block comprising a first block pointer comprising a first stored checksum and a first data block location and a second block pointer comprising a second stored checksum and a second data block location;
 - obtaining the first data block using the first data block location;
 - calculating the checksum for the first data block to obtain a calculated checksum;
 - retrieving the data from the first data block, if the stored checksum equals the calculated checksum; and
 - performing an appropriate action, if the first stored checksum is not equal to the calculated checksum.
16. (Original) The method of claim 15, wherein the calculated checksum is calculated using a storage pool allocator.
17. (Currently Amended) A method for storing and retrieving [[a]] data blocks, comprising:
- storing [[the]] a first data block and a second data block;
 - obtaining a first data block location and a second data block location;
 - calculating a first data block checksum for the first data block and a second data block checksum for the second data block;
 - storing a first block pointer in an indirect block, wherein the first block pointer comprises the first data block location and the first data block checksum;
 - storing a second block pointer in the indirect block, wherein the second block pointer comprises the second data block location and the second block checksum;
 - obtaining the indirect block comprising the first block pointer and the second block pointer;
 - obtaining the first data block using the first data block location stored in the first block pointer;
 - calculating the checksum for the first data block to obtain a calculated checksum;
 - retrieving data from the first data block, if the first data block checksum stored in the first block pointer equals the calculated checksum; and
 - performing an appropriate action, if the first data block checksum is not equal to the first calculated checksum.

18. (Currently Amended) A system for storing [[a]] data blocks, comprising:
 - a storage pool comprising [[the]] a first data block, a second data block and a first indirect block referencing the first data block and the second data block, wherein the first indirect block comprises a first data block checksum and a first data block location stored in a first block pointer, and a second data block checksum and a second data block location stored in a second block pointer; and
 - a storage pool allocator configured to store the first data block, the second data block and the first indirect block in the storage pool.
19. (Original) The system of claim 18, further comprising:
 - a second indirect block, comprising a first indirect data block checksum and a first indirect block location,
 - wherein the storage pool allocator is further configured to store the second indirect block in the storage pool.
20. (Original) The system of claim of claim 19, further comprising:
 - a data management unit configured to assemble the first indirect block and request the storage pool allocator to store the first indirect block.
21. (Original) The system of claim 20, wherein the storage pool comprises at least one storage device.
22. (Original) The system of claim 20, wherein the storage pool is divided into a plurality of metaslabs.
23. (Original) The system of claim 22, wherein each of the plurality of metaslabs is associated with a metaslab ID.
24. (Currently Amended) The system of claim 23, wherein the first data block location comprises the metaslab ID and an offset.
25. (Currently Amended) A computer system for storing [[a]] data blocks, comprising:
 - a processor;
 - a memory;
 - a storage device; and

software instructions stored in the memory for enabling the computer system under control of the processor, to:

store [[the]] a first data block and a second data block in a storage pool;

obtain a first data block location and a second data block location;

calculate a first data block checksum for the first data block;

calculate a second data block checksum for the second data block; and

store a first indirect block referencing the first data block and the second data block in the storage pool, wherein the first indirect block comprises a first block pointer comprising the first data block location and the first data block checksum and a second block pointer comprising the second data block location and the second data block checksum.

26. (Currently Amended) A network system having a plurality of nodes, comprising:

a storage pool comprising [[the]] a first data block, a second data block and a first indirect block referencing the first data block and the second data block, wherein the first indirect block comprises a first data block checksum and a first data block location stored in a first block pointer, and a second data block checksum and a second data block location stored in a second block pointer; and

a storage pool allocator configured to store the first data block, the second data block and the first indirect block in the storage pool,

wherein the storage pool is located on any one of the plurality of nodes, and

wherein the storage pool allocator is located on any one of the plurality of nodes.